CS380: Introduction to Computer Graphics

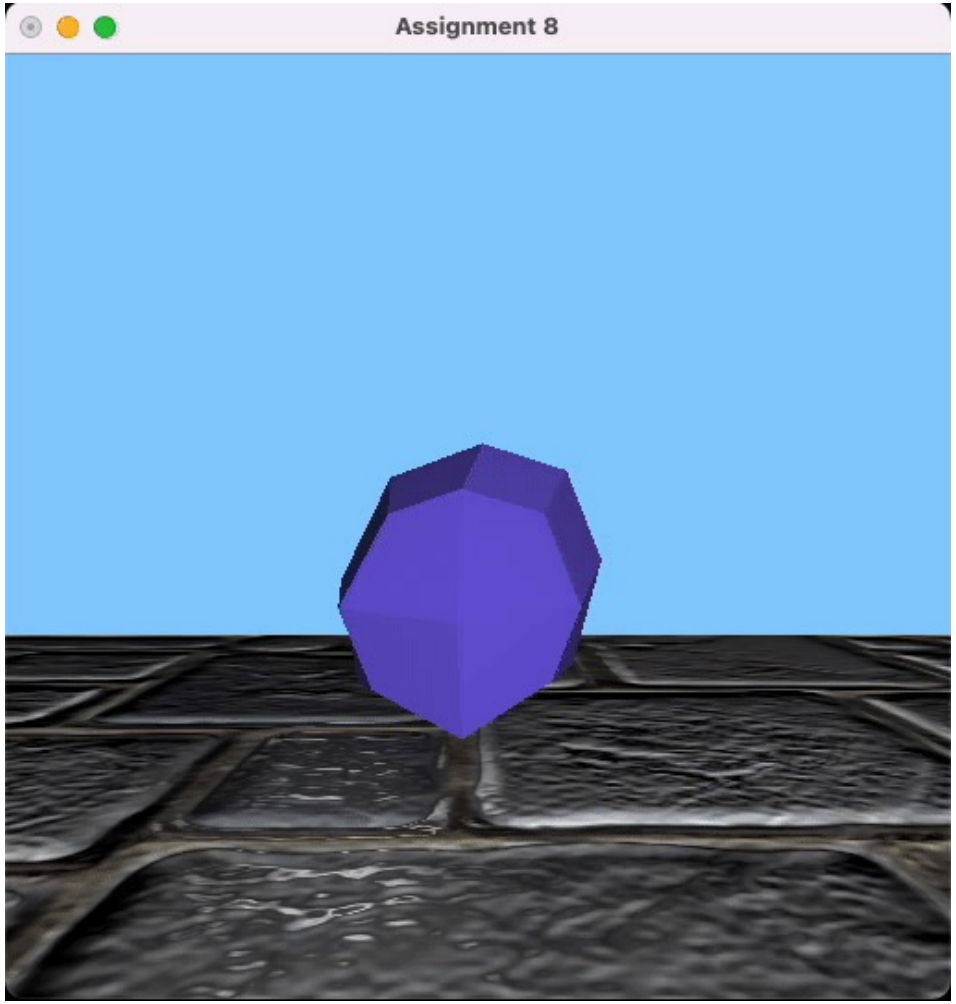
Meshes and Subdivision

LAB SESSION 8
KUNHO KIM

Spring 2023
KAIST

Overview

- Smooth shading
  - Normal selection
- Mesh subdivision
  - Catmull-Clark subdivision



Goals

- **Task 1:** Read the mesh file "`cube.mesh`".
- **Task 2:** Implement smooth shading.
- **Task 3:** Animate the vertices of the cube.
- **Task 4:** Implement subdivision.
- **Task 5:** Make your own mesh file "`model.mesh`".

Preparation

- Assignment 8 will be built upon your **assignment 7** codebase.
- Download the assignment 8 code files and integrate them with your **assignment 7** codebase.

Preparation

These are the new files:

- “cube.mesh”
- “mesh.h”
- “mesh.interface”
- “shaders/specular-gl2.fshader”
- “shaders/specular-gl3.fshader”

Preparation

We provide a file “`mesh.interface`” which documents the interface provided by the `Mesh` class, and a sample usage of the `VertexIterator` class.

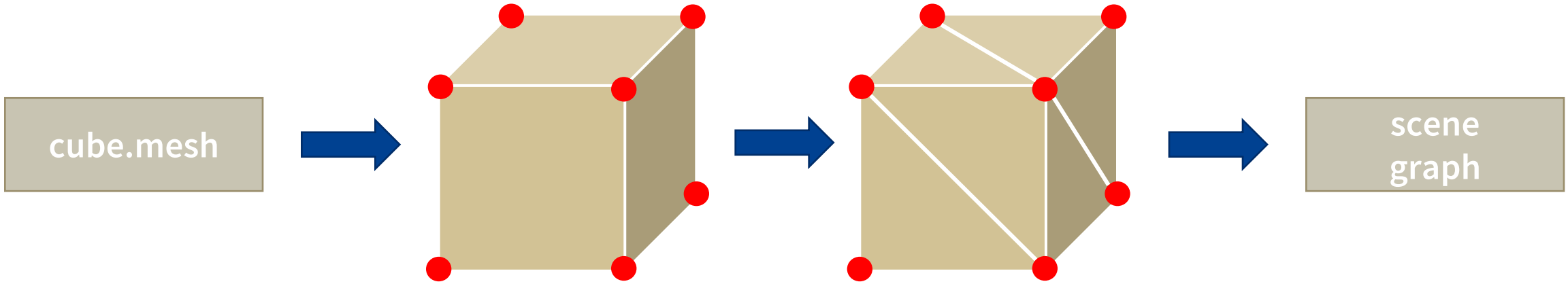
```
// Sample usage for VertexIterator:

for (int i = 0; i < mesh.getNumVertices(); ++i) {
  const Mesh::Vertex v = mesh.getVertex(i);

  Mesh::VertexIterator it(v.getIterator()), it0(it);
  do
  {
    [...]                                   // can use here it.getVertex(), it.getFace()
  }
  while (++it != it0);                      // go around once the 1ring
}
```

Preparation

- There is also a new fragment shader: “`specularg1{2|3}.fshader`”.
- You should pair it up with the basic vertex shader and load them into a new `Material`.

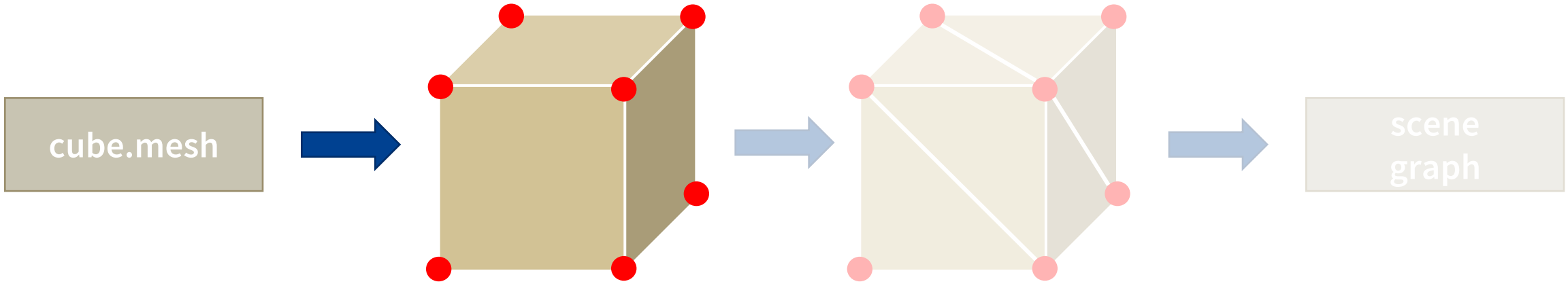
Task 1

1. Read the "`cube.mesh`" file.
2. Convert each quad into two triangles before loading it into the `SimpleGeometryPN` object.
3. Add a new `SgRbtNode` to the scene graph.



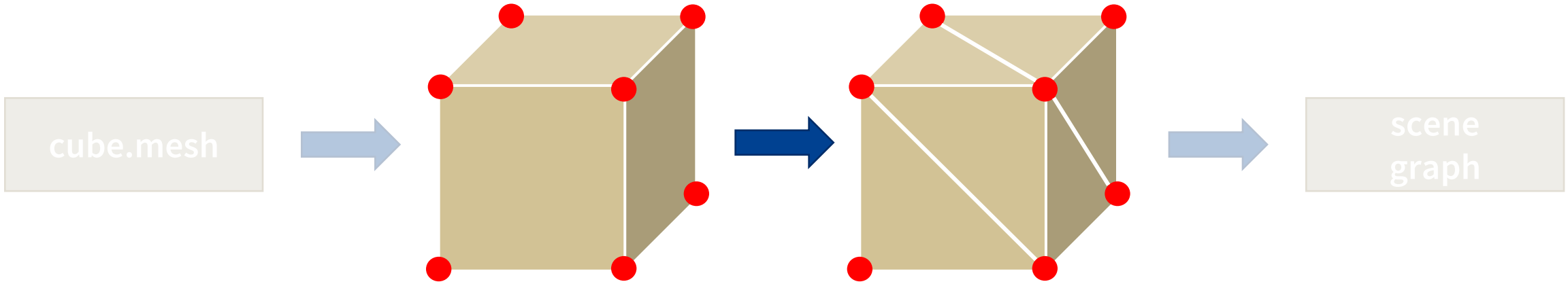
Task 1

1. Read the "`cube.mesh`" file.
   - Use "`Mesh->load(const char filename[])`" to read mesh.



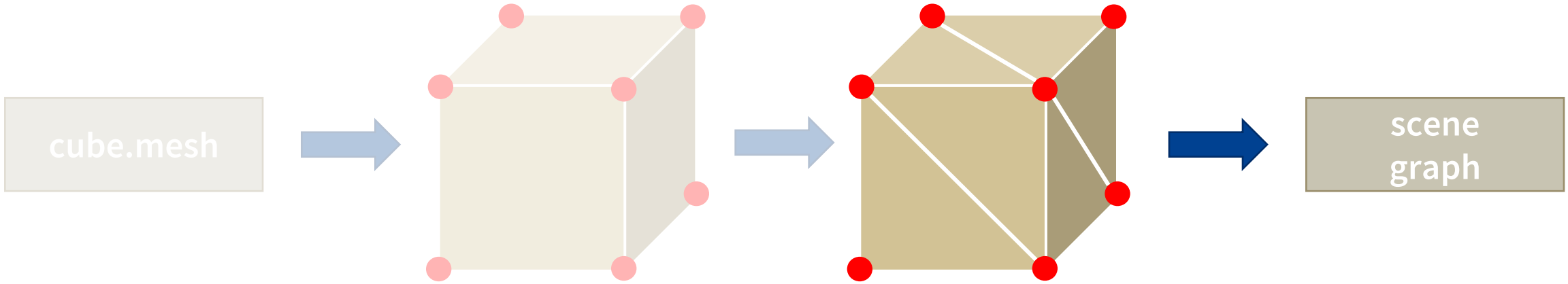
Task 1

2. Convert each quad into two triangles before loading it into the `SimpleGeometryPN` object.
   - Upload vertices and indices to an existing `SimpleGeometryPN` object.
   - Use a member function `upload()`.

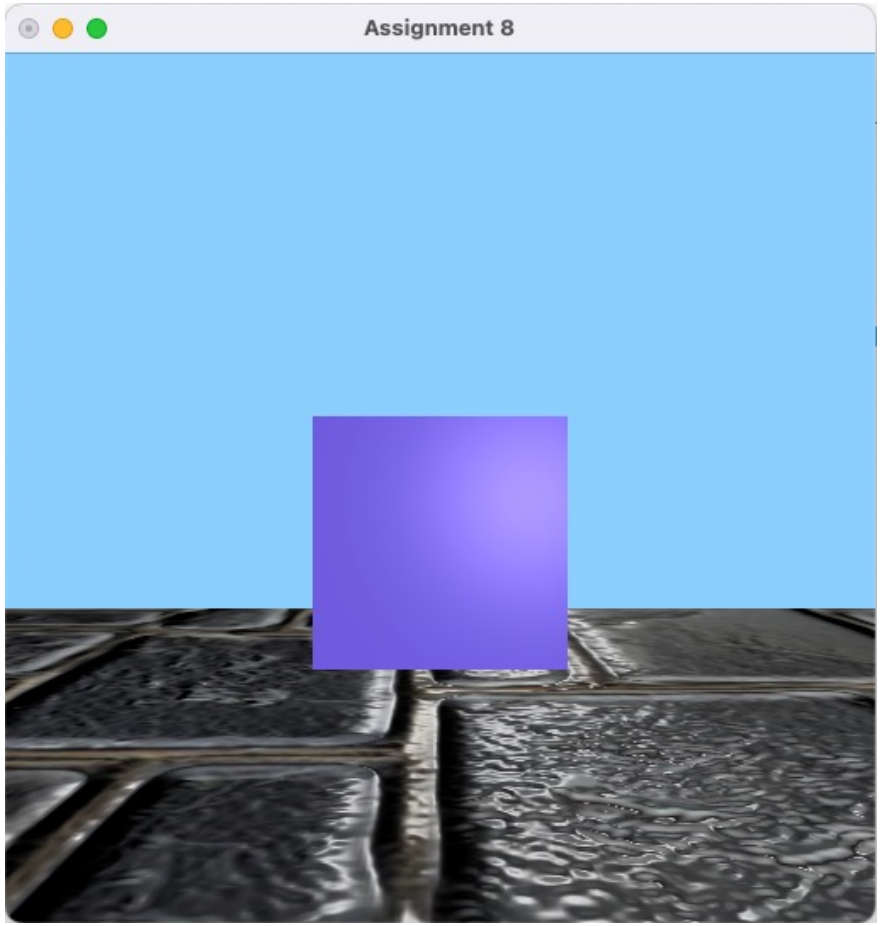


Task 1

3. Add a new `SgRbtNode` to the scene graph.
   - Add a new specular `Material` in `initMaterials()`.
   - Add a new child node in `initScene()`.



Result After Task 1



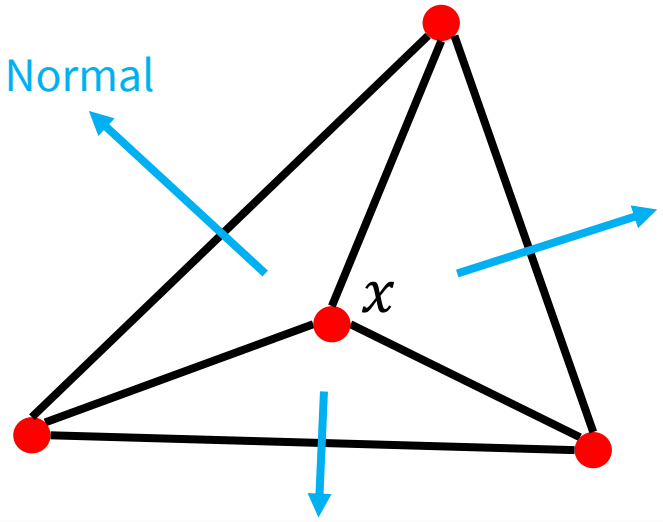
Task 2

- Flat shading

  The normal of a vertex is the same as the normal of the face to which the vertex belongs.

- Smooth shading

  The normal of the vertex is the average of the normals of all incident faces.



What is the normal of x?

Task 2

- For smooth shading, you can use `VertexIterator`, `Vertex.getNormal()`, and `Vertex.setNormal()` to compute the average normal.
- If we set the vertex normal and `Material` correctly, "`specularg1{2|3}.fshader`" automatically applies the flat or smooth shading.

Task 2

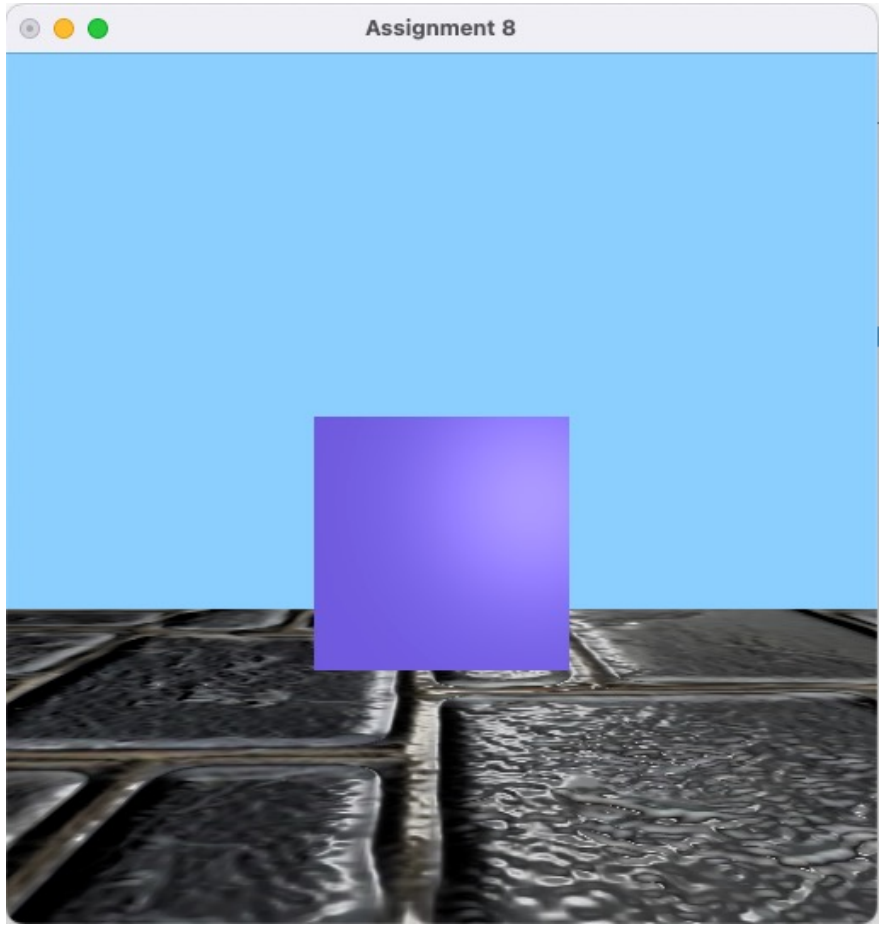
To implement smooth shading, you can follow these steps:

1. Zero out all of the vertex normals.

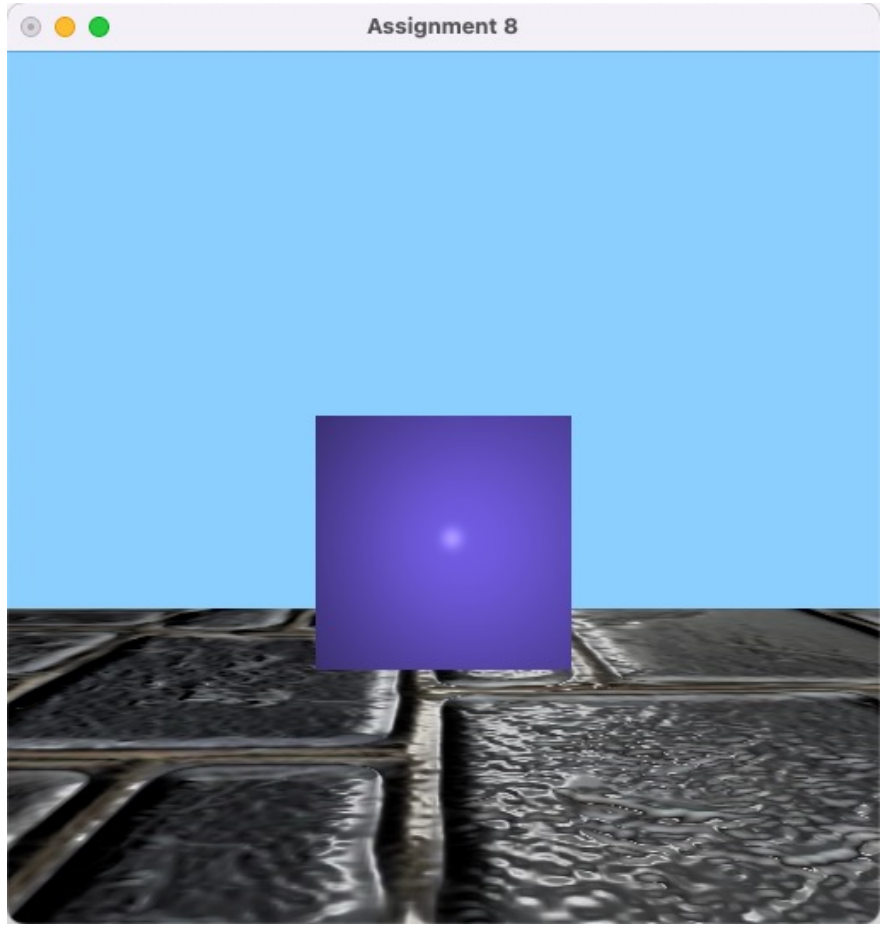
2. Iterate through the faces of the mesh and accumulate its normal to all of its surrounding vertices.

3. Visit each vertex and normalize the vertex normal.

Result After Task 2



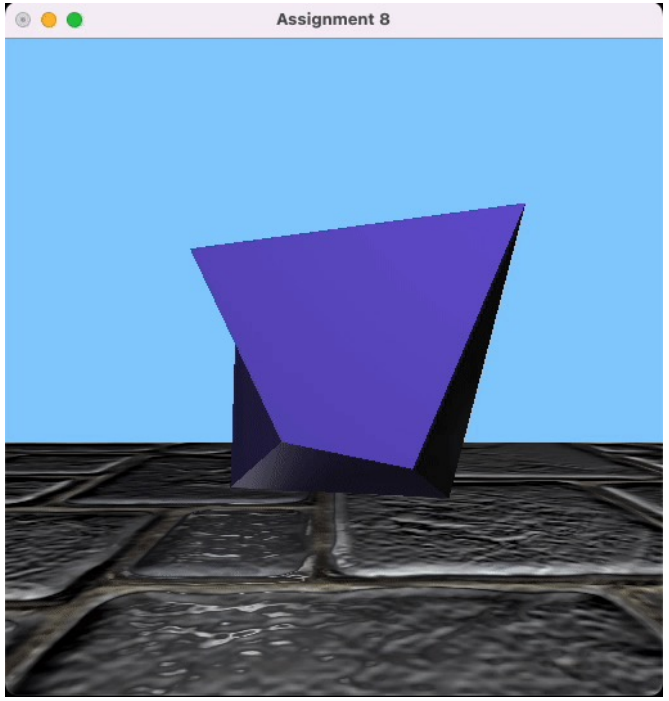
Press ‘f’



Task 3

A new callback for animating the vertices of the cube.

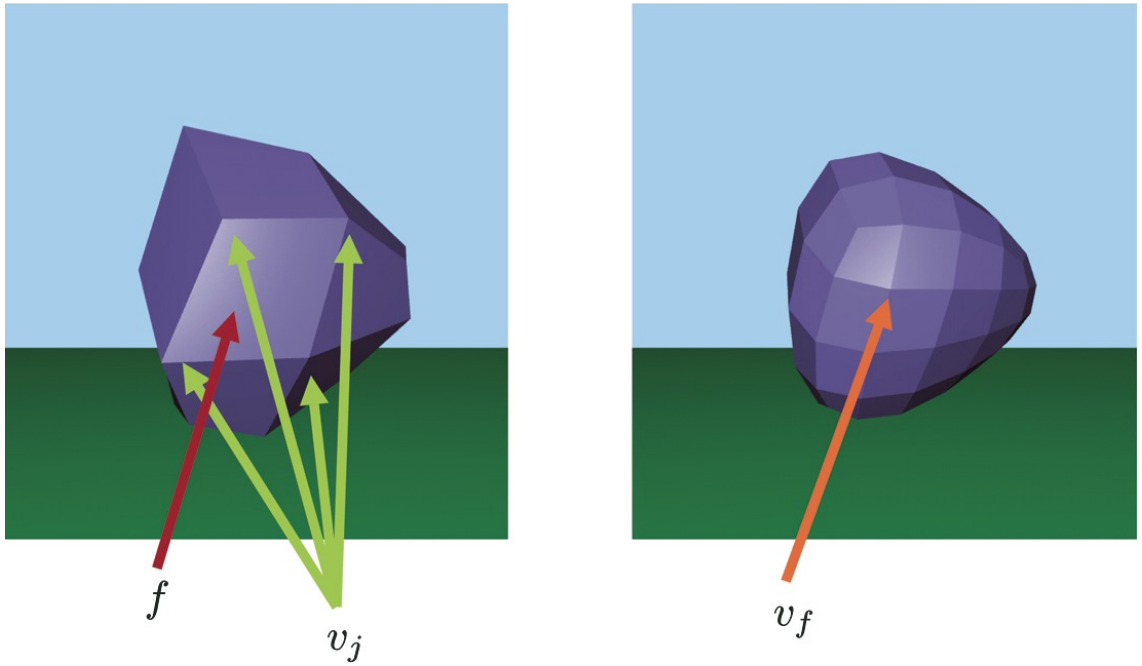
- Use GLUT timer callback (as in the previous assignments).
- Since the cube's vertices are animated, we should dump them into the geometry.



Task 4

Subdivision with Catmull-Clark algorithm.

Face-vertex $v_f = \frac{1}{m_f}\sum_j v_j$ (Centroid of vertices) m_f: the number of vertices

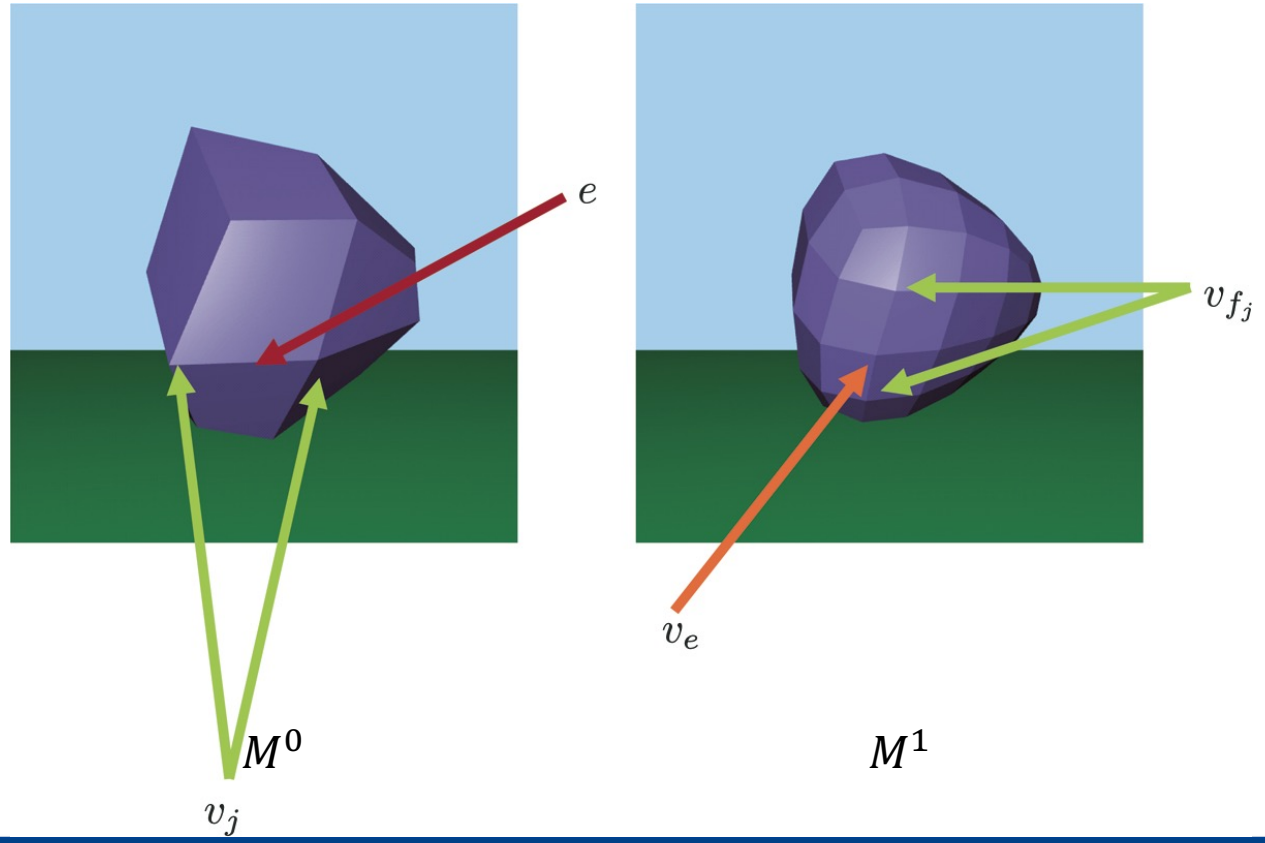


Watertight mesh M^0 M^1

Task 4

Subdivision with Catmull-Clark algorithm.

Edge-vertex $v_e = \frac{1}{4}(v_1 + v_2 + v_{f_1} + v_{f_2})$

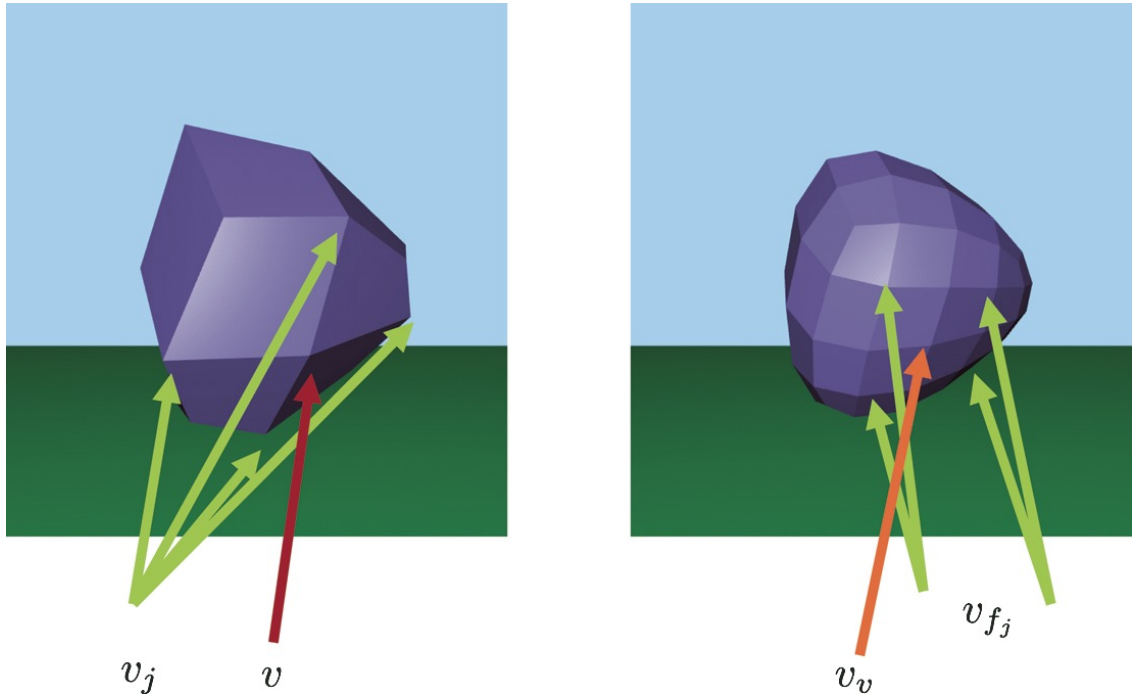


Watertight mesh

Task 4

Subdivision with Catmull-Clark algorithm.

Vertex-vertex $v_v = \frac{n_v - 2}{n_v} v + \frac{1}{n_v^2} \sum_j v_j + \frac{1}{n_v^2} \sum_j v_{f_j}$ n_v: the number of connected vertices



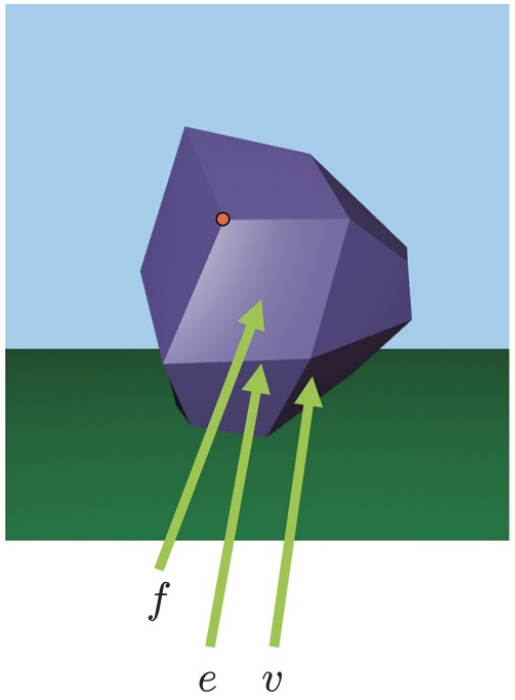
Watertight mesh M^0 M^1

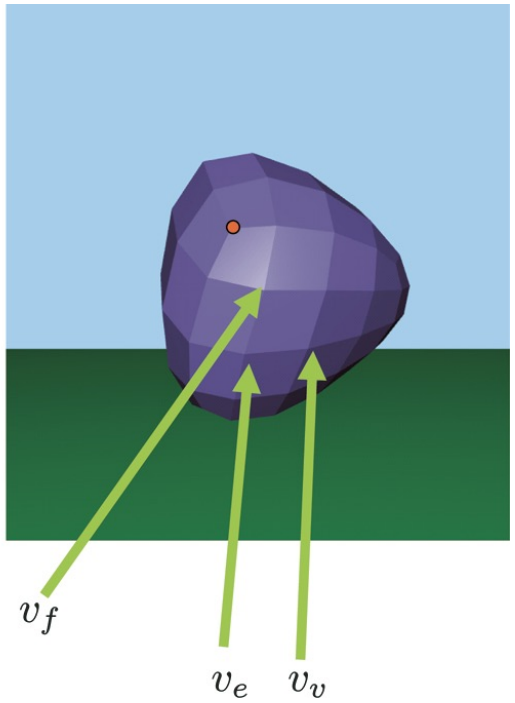
Task 4

Face-vertex $v_f = \frac{1}{m_f}\sum_j v_j$

Edge-vertex $v_e = \frac{1}{4}(v_1 + v_2 + v_{f_1} + v_{f_2})$

Vertex-vertex $v_v = \frac{n_v - 2}{n_v} v + \frac{1}{n_v^2}\sum_j v_j + \frac{1}{n_v^2}\sum_j v_{f_j}$





Task 4

- Face-vertices: New vertices inserted in a face.
- Edge-vertices: New vertices inserted on an edge.
- Vertex-vertices: Updated vertices of the old vertices.
- See the details in the textbook “22.6.1 Catmull-Clark” or lecture notes 12-2.

Task 4

To implement subdivision, you can follow these steps:

1. Loop over all of the faces of the mesh and compute face-vertex values, then call `setNewFaceVertex()`.
2. Loop over all of the edges and compute edge-vertex values, then call `setNewEdgeVertex()`.
3. Loop over all of the vertices and compute vertex-vertex values, then call `setNewVertexVertex()`.
4. Once all the slots are filled in, call `subdivide()`.

Task 4

Your final version will include the following hot keys:

- ‘f’: toggle between smooth and at shading.
- ‘0’: increase the number of subdivision steps applied to the cube before being drawn. The upper limit of subdivision steps is 6.
- ‘9’: decrease the number of subdivision steps applied to the cube before being drawn.
- ‘8’: double the speed at which the cube deforms.
- ‘7’: half the speed at which the cube deforms.

Task 5

Make your own mesh file “`model.mesh`”.

- The number of faces for the mesh must be equal or greater than 30.
- Your new mesh must be watertight and visible entirely on the start screen.
- The new mesh must be loaded with the same mesh loader function.
- You will get a zero score for this task if you violate any condition above.
- You can handcraft the model, make a script, or use any tools to make the model.

vertice를 cube에 넣는게 쉬움
glender?

Task 5

- There is the best model award. Show off your creativity!
  - Best model: 50 bonus points.
  - Honorable mentions: 30 bonus points.

- The bonus points makes up for the points you lost in the other assignments (but NOT in the exams and participation scores).

TODOs

- **Task 1:** Read the mesh file "`cube.mesh`".
- **Task 2:** Implement smooth shading.
- **Task 3:** Animate the vertices of the cube.
- **Task 4:** Implement subdivision.
- **Task 5:** Make your own mesh file "`model.mesh`".

Evaluation

- Is there a cube that read from a file in the scene?
- Is the smooth shading implemented correctly when pressing the hotkey ‘f’?
- Does the speed change correctly when pressing the hotkeys ‘7/8’?
- Does the subdivision change correctly when pressing the hotkeys ‘0/9’?
- Does “`model.mesh`” open correctly?

Submission

- Due: Sun, Jun 11 23:59 KST.
- No late submission for this assignment.
- Compress the codes in a .zip file and submit it on Gradescope.